MarshallSoft Client Mailer Library for Visual dBASE

Reference Manual

(MCM4DB)

Version 5.0

October 29, 2014

This software is provided as-is.
There are no warranties, expressed or implied.

Copyright (C) 2014
All rights reserved

MarshallSoft Computing, Inc.
Post Office Box 4543
Huntsville AL 35815

Email: info@marshallsoft.com
Web: www.marshallsoft.com

MARSHALLSOFT is a registered trademark of MarshallSoft Computing.

TABLE OF CONTENTS

1 Introduction Page 3
1.1 General Remarks Page 3
1.2 MCM Files Page 3
1.3 Documentation Set Page 4
2 Library Overview Page 5
2.1 mcmAttach Page 5
2.2 mcmGetError Page 6
2.3 mcmGetInteger Page 7
2.4 mcmGetInteger2 Page 9
2.5 mcmGetLetterMacro Page 10
2.6 mcmGetListMacro Page 11
2.7 mcmGetString Page 12
2.8 mcmGetString2 Page 13
2.9 mcmLoadString Page 14
2.10 mcmMergeNext Page 15
2.11 mcmMergeText Page 16
2.12 mcmOpenBounce Page 17
2.13 mcmOpenHeader Page 18
2.14 mcmOpenLetter Page 19
2.15 mcmOpenList Page 20
2.16 mcmOpenReply Page 21
2.17 mcmOpenSkip Page 22
2.18 mcmPop3Close Page 23
2.19 mcmPop3Connect Page 24
2.20 mcmReadReply Page 25
2.21 mcmRelease Page 26
2.22 mcmSendMail Page 27
2.23 mcmSetInteger Page 28
2.24 mcmSetProxySSL Page 29
2.25 mcmSetString Page 30
2.26 mcmSleep Page 31
2.27 mcmSmtpClose Page 32
2.28 mcmSmtpConnect Page 33
2.29 mcmStatistics Page 34
2.30 mcmUtility Page 35
2.31 mcmWriteToLog Page 36
3 MCM Error List Page 37

1 Introduction

The **MarshallSoft Client Mailer for Visual dBASE** provides the capability to send **personalized** email to clients or customers **directly** from a dBASE or dBASE Plus application program. The "MarshallSoft Client Mailer for Visual dBASE Reference Manual" contains details on each individual MCM function.

The most current version of the **MarshallSoft Client Mailer for Visual dBASE** can be found at http://www.marshallsoft.com/mcm4db.htm

1.1 General Remarks

This is the Reference Manual for the Visual dBASE/dBASE Plus version of the MarshallSoft Client Mailer (MCM). Other versions are for C/C++, Delphi, Visual Basic, Visual FoxPro, and Xbase++. All versions employ the identical MCM32.DLL differing only in documentation and example programs.

All MarshallSoft Client Mailer (MCM) functions return an integer code. Negative values are always errors. Refer to Section 3.0 below, "MCM Error Return Code List". The file mcmErrors.txt contains a list of all error codes and their corresponding numerical value.

Non-negative return codes are never errors. Note that the **mcmErrorText** function is used to get the text message associated with any error code.

1.2 MCM Files (Visual dBASE)

- `mcm32.cc` `MCM function declaration & constants file.`
- `mcm32.dll` `32-bit MarshallSoft Client Mailer DLL.`
- `*.prg` `dBASE example programs.`

1.3 Documentation Set

There are five manuals in Adobe PDF format for the **MarshallSoft Client Mailer.**

- **Tutorial Manual**: Introduces the basic functionality and overview of the **MarshallSoft Client Mailer.**
- **Servers Manual**: Covers background information on SMTP & POP3 servers.
- **User Manual**: Covers information that is not programming language specific (letter & list preparation, program logic, purchasing, performance, SSL, etc.).
- **Reference Manual**: Contains details for each individual MCM function specific for each programming language (dBASE, Visual FoxPro, Delphi, C/C++, VB, etc.).
- **Programmer Manual**: Contains programming language (dBASE, Visual FoxPro, Delphi, C/C++, etc.) specific information such as compiling and running example programs.

The manuals can be found in the DOCS subdirectory/folder in the **MarshallSoft Client Mailer** file structure when it is installed. Note that for clarity purposes some information may have been provided in more than one manual.

- **Tutorial Manual**: mcm_tutorial.pdf or online at http://www.marshallsoft.com/mcm_tutorial.pdf.
- **Servers Manual**: mcm_servers.pdf or online at http://www.marshallsoft.com/mcm_servers.pdf.
- **User Manual**: mcm_users.pdf or online at http://www.marshallsoft.com/mcm_users.pdf.
- **Reference Manuals** mcm4db_reference.pdf or online at http://www.marshallsoft.com/mcm4db_reference.pdf.
- **Programmer Manuals** mcm4db_programmer.pdf or online at http://www.marshallsoft.com/mcm4db_programmer.pdf

Links to the above manuals can also be found online on the **MarshallSoft Client Mailer for dBase** product page http://www.marshallsoft.com/mcm4db.htm

2.0 MCM Functions

All MCM functions are declared in file MCM32.CC

2.1 mcmAttach: Initialize MarshallSoft Client Mailer.

SYNTAX

```
CLONG mcmAttach(CLONG,CLONG,CLONG,CLONG,CSTRING)

Code = mcmAttach(KeyCode, EditionCode, ChansWanted, DebugLevel, PathToMCM)

   KeyCode     : MCM key code (identifies purchaser)
   EditionCode : MCM edition code (no longer used)
   ChansWanted : Maximum # channels requested.
   DebugLevel  : Debug level (0=OFF, 1=LOW, 2=HIGH)
   PathToMCM   : Pathname of files folder.
```

REMARKS

The **mcmAttach** function initializes the Client-Mailer DLL (MCM32.DLL), passing the initialization parameters (1) KeyCode [0 for the evaluation version], (2) Edition Code [pass 0], (3) the maximum number of channels to use when sending email, and (4) the debug level; 0 for no debug, 1 for low, and 2 for high, and (5) PathToMCM, the pathname of the log file folder.

A keycode file (keycode.cc) containing the customer's keycode is included when MCM4DB is purchased.

mcmAttach must be the first MarshallSoft Client Mailer (MCM) function called, excepting mcmUtility.

RETURNS

Evaluation: # days remaining in the evaluation (trial) period.
Purchased: 999

EXAMPLE CODE

```
KeyCode = 0
EditionCode = 0
ChansWanted = 24
PathToMCM = "c:\mcm4db\apps"
Code = mcmAttach(KeyCode,EditionCode,ChansWanted,MCM_DEBUG_OFF,PathToMCM)
```

EXAMPLE PROGRAMS

TestMCM, SendMail and GetReply

2.2 mcmGetError: Get text associated with error code.

SYNTAX

```
CLONG mcmGetError(CLONG,CSTRING,CLONG)

Code = mcmGetError(ErrCode, Buffer, BufLen)

   ErrCode : (CLONG)   Error code.
   Buffer  : (CSTRING) Error text buffer.
   BufLen  : (CLONG)   Size of buffer.
```

REMARKS

The **mcmGetError** function is used to copy the error text associated with the error code 'ErrCode' returned by a MCM function to the buffer, where it can be displayed by the calling program code.

The size of the buffer should be 256 bytes.

RETURNS

Return = 0 : No such error.
Return < 0 : The number of bytes copied into the buffer.

EXAMPLE CODE

```
Buffer = Space(256)
if ErrCode < 0
  * get MCM error message
  Code = mcmGetError(ErrCode, Buffer, 255)
  . . .
```

EXAMPLE PROGRAMS

SendMail and GetReply

2.3 mcmGetInteger: Gets MCM processing information.

SYNTAX

```
CLONG mcmGetInteger(CLONG)

Code = mcmGetInteger(ParamName)

   ParamName : (CLONG) Parameter number.
```

REMARKS

The **mcmGetInteger** function returns an integer whose value depends on the value of the passed parameter 'ParamName' as follows.

RETURNS

MCM_GET_VERSION : The version of MCM in packed hexadecimal format (X.Y.Z)

MCM_GET_VERSION_1ST_PART : The first digit of the version of MCM.

MCM_GET_VERSION_2ND_PART : The second digit of the version of MCM.

MCM_GET_VERSION_3RD_PART : The third digit of the version of MCM.

MCM_GET_BUILD : The build number of MCM.

MCM_GET_LETTER_LINE_NBR : The current letter line just processed.

MCM_GET_LETTER_CHAR_POS : The current character position on the current letter line.

MCM_GET_LETTER_MACROS : The number of macros (substitution strings) found in the letter.

MCM_GET_LIST_LINE_NBR : The current list line just processed.

MCM_GET_MAX_LIST_SIZE : The maximum number of entries allowed in the list of recipients.

MCM_GET_MAX_CHANNELS : The number of channels being used to send email.

MCM_GET_CUSTOMER_ID: The customer ID.

MCM_GET_ALLOWED_CHANNELS: The maximum allowed number of channels.

MCM_GET_ALLOWED_LIST_SIZE: The maximum allowed list size.

MCM_GET_ALLOWED_SKIP_FILES: The maximum allowed number o skip files.

MCM_GET_ALLOWED_REPLY_FILES: The maximum number of reply files.

MCM_GET_EDITION: The MCM edition (no longer used).

MCM_GET_REGISTRATION: The customer registration string.

MCM_GET_CHANNEL_STATUS : The current channel status where each bit represents one channel.

MCM_GET_EMAIL_QUEUED_COUNT : The number of emails queued to be sent.

MCM_GET_EMAIL_SENT_COUNT : The number of emails successfully sent.

MCM_GET_EMAIL_ERROR_COUNT : The number of emails queued but not sent due to errors.

MCM_GET_LIST_LINES : The number of lines in the list file.

MCM_GET_LETTER_LINES : The number of lines in the letter file.

MCM_GET_SKIP_LINES : The number of lines in last skip file loaded.

MCM_GET_LIST_MACRO_COUNT : The number of macros (substitution strings) in the recipient list.

MCM_GET_LIST_DELIMITER : The macro (substitution string) delimiter. This will be either the comma, semicolon, tab, carrot ^, or tilde ~.

MCM_GET_LIST_ERROR_CSTRING : The line number of last error in the recipient list.

MCM_GET_SKIP_FILE_LIMIT : The maximum number of skip files allowed.

MCM_GET_REPLY_FILE_LIMIT : The maximum number of reply files allowed.

EXAMPLE CODE

```
? "Customer ID is " + Str(mcmGetInteger(MCM_GET_CUSTOMER_ID))
```

EXAMPLE PROGRAM

SendMail

2.4 mcmGetInteger2: Get information for macro processing.

SYNTAX

```
CLONG mcmGetInteger2(CLONG,CLONG)

Code = mcmGetInteger2(ParamName, Select)

   ParamName : (CLONG) Parameter number.
   Select    : (CLONG) Selection number for ParamName.
```

REMARKS

The **mcmGetInteger2** function returns an integer value corresponding to the passed parameters 'ParamName' and 'Select'.

RETURNS

MCM_GET_LETTER_MACRO_LINE : Get the line on which the 'Select' macro appears.

EXAMPLE CODE

```
* get macro string i (1,2,3,...)
Buffer = Space(256)
Code = mcmGetLetterMacro(i, Buffer, 255)
if Code > 0
  * find line in letter on which macro # i occurs
  Code = mcmGetInteger2(MCM_GET_LETTER_MACRO_LINE, i)
  ? "Macro " + Buffer + " defined on line " + Str(Code)
```

EXAMPLE PROGRAMS

(none)

2.5 mcmGetLetterMacro: Get Macro Substitution String in Letter

SYNTAX

```
CLONG mcmGetLetterMacro(CLONG,CSTRING,CLONG)

Code = mcmGetLetterMacro(MacroNumber, Buffer, BufSize )

   MacroNumber : (CLONG)   Macro number (1,2,...)
   Buffer      : (CSTRING) Macro buffer.
   BufSize     : (CLONG)   Size of buffer.
```

REMARKS

The **mcmGetLetterMacro** returns the macro (substitution string) in the letter associated with the macro number (1,2,3,...). Macros may be up to 40 characters in length. The first macro in a letter is #1, the second is #2, etc.

For example, consider the letter as shown in section 2.12 **mcmOpenLetter**. The first macro in the letter is *"%EmailAddress%*, the second is *%Fullname%*, etc.

RETURNS

Return > 0 : The line number (in the letter) on which macro appears.
Return < 0 : The error code MCM_NO_SUCH_MACRO.

EXAMPLE CODE

```
* get macro string i (1,2,3,...)
Buffer = Space(256)
Code = mcmGetLetterMacro(i, Buffer, 255)
if Code > 0
  * find line in letter on which macro # i occurs
  Code = mcmGetInteger2(MCM_GET_LETTER_MACRO_LINE, i)
  ? "Macro " + Buffer + " defined on line " + Str(Code)
```

EXAMPLE PROGRAMS

(none)

ALSO SEE

mcmGetListMacro

2.6 mcmGetListMacro: Get Macro Substitution String in Recipient List

SYNTAX

```
CLONG mcmGetListMacro(CLONG,CSTRING,CLONG)

Code = mcmGetListMacro(MacroNumber, Buffer, BufSize)

   MacroNumber : (CLONG)   Macro number (1,2,...)
   Buffer      : (CSTRING) Macro buffer.
   BufSize     : (CLONG)   Size of buffer.
```

REMARKS

The **mcmGetListMacro** returns the macro (substitution string) in the recipient list associated with the macro number (1,2,3,...). Macros may be up to 40 characters in length, and are defined on the first line of the list.

For example, consider the recipient list as shown in section 2.13 **mcmOpenList**. There are three macros (always appearing on the first line) in the list. The first macro is *%EmailAddress%*, the second is *%AppointmentTime%*, and the third is *%Fullname%*.

mcmGetListMacro is called by the application code that sends the email such as the **SendMail** example program.

RETURNS

Return > 0 : The macro index.
Return < 0 : The error code MCM_NO_SUCH_MACRO.

EXAMPLE CODE

```
* get list macro i (1,2,3,...)
Buffer = Space(256)
Code = mcmGetListMacro (i, Buffer, 255)
if Code > 0
  ? "Macro = " + Buffer
```

EXAMPLE PROGRAMS

(none)

ALSO SEE

mcmGetLetterMacro

2.7 mcmGetString: Gets string parameter for MCM processing.

SYNTAX

```
CLONG mcmGetString(CLONG,CSTRING,CLONG)

Code = mcmGetString(ParamName, Buffer, BufSize)

   ParamName : (CLONG)   Parameter number
   Buffer    : (CSTRING) String buffer.
   BufSize   : (CLONG)   Size of buffer.
```

REMARKS

The **mcmGetString** function returns a string which contents depends on the value of the passed parameter 'ParamName' as follows. Note that MCM reads only the headers of incoming email.

MCM_GET_VERSION : Copies the MCM version string into 'Buffer'.

MCM_GET_LETTER : Copies the entire letter into 'Buffer'. Requires SMTP connection.

MCM_GET_SUBJECT : Copies the letter subject into 'Buffer'. Requires SMTP connection.

MCM_GET_BODY : Copies the body of the letter into 'Buffer'. Requires SMTP connection.

MCM_GET_FROM : Copies the "From:" address into 'Buffer'. Requires SMTP connection.

MCM_GET_TIME_STAMP : Copies the current date & time string into 'Buffer'.

MCM_GET_LAST_EMAIL_SENT : Copies address of last email sent into 'Buffer'.

MCM_GET_RECIPIENT : Copies email address of last recipient.

RETURNS

The number of characters copied.

EXAMPLE CODE

```
Work = Space(4096)
Code = mcmGetString(MCM_GET_LETTER, Work, 4095)
if Code > 0
  ? Work
```

EXAMPLE PROGRAMS

SendMail and GetReply

ALSO SEE

mcmGetInteger and mcmGetInteger2

2.8 mcmGetString2: Gets string parameter for MCM processing.

SYNTAX

```
Code = XmcmGetString2(ParmName, Selection, Buffer, BufSize)

   ParmName : (CLONG) Parameter number
   Selection: (CLONG) Selection index (1,2,...)
   Buffer   : (CSTRING) String buffer.
   BufSize  : (CLONG) Size of buffer.
```

REMARKS

The **mcmGetString2** function returns a string which contents depends on the value of the passed parameter 'ParamName' as follows. Note that MCM reads only the headers of incoming email.

`MCM_GET_BOUNCE_STRING` : Copies the selected 'bounce' string into 'Buffer'. The bounce string must have been previously set by mcmSetString(MCM_ADD_BOUNCE_STRING, String).

Bounce strings are numbered 1,2,...

RETURNS

The number of characters copied.

EXAMPLE CODE

```
Work = Space(256)
Code = mcmGetString(MCM_GET_BOUNCE_STRING, 1, Work, 255)
if Code > 0
```

EXAMPLE PROGRAMS

GetReply

ALSO SEE

mcmGetString, mcmGetInteger, and mcmGetInteger2

2.9 mcmLoadString: Load substitution string.

SYNTAX

```
CLONG mcmLoadString(CSTRING,CSTRING)

Code = mcmLoadString(StringKey, StringText)

   StringKey  : (CSTRING) String key.
   StringText : (CSTRING) String text.
```

REMARKS

The **mcmLoadString** loads the substitution string for advanced macros.

Note: Advanced macros have not been implemented yet.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : No error

EXAMPLE CODE

(none)

EXAMPLE PROGRAMS

(none)

2.10 mcmMergeNext: Merge next recipient for sending.

SYNTAX

```
CLONG mcmMergeNext(CVOID)

Code = mcmMergeNext()
```

REMARKS

The **mcmMergeNext** function merges the next recipient from the recipient list with the loaded letter in preparation for sending.

See the **mcmMergeText** function for a list of the merge codes.

mcmMergeNext is called by the application code that sends the email such as the **SendMail** example program.

RETURNS

< -1 Error (see Section 3.0 MCM Error List)
= -1 End-of-file (MCM_EOF)
= 0 OK to send
> 0 Don't send (see mcmMergeText)

EXAMPLE CODE

```
* send letter to each recipient
for I = 1 To 1000
  * merge letter with next recipient
  MergeCode = mcmMergeNext()
```

EXAMPLE PROGRAMS

GetReply

ALSO SEE

mcmMergeText

2.11 mcmMergeText: Get text for associated merge code.

SYNTAX

```
CLONG mcmMergeText(CLONG,CSTRING,CLONG)

Code = mcmMergeText(MergeCode, Buffer, BufSize)

   MergeCode : (CLONG)   Merge code.
   Buffer    : (CSTRING) String buffer.
   BufSize   : (CLONG)   Size of buffer.
```

REMARKS

The **mcmMergeText** function copies the merge code text corresponding with the numerical 'MergeCode' to 'Buffer' so that it can be displayed by the calling application program.

Recall that if the value returned by the **mcmMergeNext** function (called the "merge code") is positive, then email should not be sent to this particular recipient. The numerical values of the merge codes are listed in mcm32.cc and include

MCM_MERGE_INVALID_ADDRESS : Invalid email address
MCM_MERGE_DUPLICATE_ADDRESS : Duplicate email address
MCM_MERGE_BRACKETS_NOT_ALLOWED : '<' and '>' not allowed in email address
MCM_MERGE_CANNOT_OPEN_ATTACH : Cannot open attachment
MCM_MERGE_UNKNOWN_CHARSET : Unknown character set
MCM_MERGE_EMTY_MACRO_CSTRING : Empty macro string found in recipient list.

In addition, merge codes between 1 and 24 indicate that the email address was found in a skip (exclusion) list:

MergeCode = 1 : Email address was found in skip list #1
. . .
MergeCode = 24 : Email address was found in skip list #24

mcmMergeText is called by the application code that sends the email such as the **SendMail** example program.

RETURNS

Number of characters copied to 'Buffer'.

EXAMPLE CODE

```
Buffer = Space(256)
if MergeCode > 0
  Code = mcmMergeText(MergeCode, Buffer, 255)
  ? Buffer
```

EXAMPLE PROGRAMS

GetReply

ALSO SEE

mcmMergeNext and mcmOpenSkip

2.12 mcmOpenBounce: Open bounce file for processing.

SYNTAX

```
CLONG mcmOpenBounce(CSTRING)

Code = mcmOpenBounce(PathName)

   PathName : (CSTRING) Pathname of bounce file.
```

REMARKS

The **mcmOpenBounce** opens the "bounce" file into which are written (when checking for client replies) the email addresses that have been returned as undeliverable (bounced).

This file is created when reading replies (see the **GetReply.prg** example program) after previously sending email (see the **SendMail.prg** example program) and can be used as one of the "skip files" the next time email is sent.

mcmOpenBounce is called by the application code that reads replies to previously sent email, such as the **GetReply** example program.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : Bounce file successfully opened.

EXAMPLE CODE

```
BounceFile = "c:\mcm4db\apps\bounce.txt"
Code = mcmOpenBounce(BounceFile)
```

EXAMPLE PROGRAMS

GetReply

ALSO SEE

mcmOpenSkip

2.13 mcmOpenHeader: Open header file for processing.

SYNTAX

```
CLONG mcmOpenHeader(CSTRING)

Code = mcmOpenHeader(HeaderName)

   HeaderName : (CSTRING) Filename of letter header file.
```

REMARKS

The **mcmOpenHeader** file opens the letter header file and scans for macros. An example of a header file is:

```
To: %EmailAddress%
Subject: Your Dental Appointment
```

Required headers are:

```
To:        Email recipient
Subject:   Email subject
```

Optional header lines are:

```
CharSet:   Character set
CC:        Carbon copy recipients
BCC:       Blind carbon copy recipients
Attach:    List of attachments
Header:    User specified SMTP header
```

Refer to the MCM User's manual (mcm_users.pdf) for details of all headers.

mcmOpenHeader is called by the application code that sends the email such as the **SendMail** example program.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : Letter file successfully opened.

EXAMPLE CODE

```
HeaderFile = "c:\mcm4db\apps\letter.hdr"
Code = mcmOpenHeader(HeaderFile)
```

EXAMPLE PROGRAMS

SendMail

ALSO SEE

mcmOpenLetter

2.14 mcmOpenLetter: Open letter file for processing.

SYNTAX

```
CLONG mcmOpenLetter(CSTRING)

Code = mcmOpenLetter(LetterPathName)

   LetterPathName : (CSTRING) Pathname of letter file.
```

REMARKS

The **mcmOpenLetter** file opens the (text or HTML) letter file, and scans the letter for macros. An example of a letter is:

```
   To: %EmailAddress%
   Subject: Your Dental Appointment

   Dear %FullName%,

   Your dental appointment is tomorrow at %AppointmentTime%.

   Sincerely,
   Dr. John H. Holliday

   PS: If you prefer that email notices not be sent, reply
   to this email with subject "REMOVE %EmailAddress%"
```

mcmOpenLetter is called by the application code that sends the email such as the **SendMail** example program.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : Letter file successfully opened.

EXAMPLE CODE

```
LetterFile = "c:\mcm4db\apps\letter.txt"
Code = mcmOpenLetter(LetterFile)
```

EXAMPLE PROGRAMS

SendMail

ALSO SEE

mcmOpenList

2.15 mcmOpenList: Open recipient list file for processing.

SYNTAX

```
CLONG mcmOpenList(CSTRING)

mcmOpenList(ListPathName)

   ListPathName : (CSTRING) Pathname of (recipient) list file.
```

REMARKS

The **mcmOpenList** file opens the recipient list file, the first line of which contains the macro substitution string. For example,

```
EmailAddress,              AppointmentTime,  FullName
m.marshall0610@yahoo.com, 10:00 am,         Mike Marshall
p.marshall0610@yahoo.com, Noon,             Paula Marshall
l.marshall0610@yahoo.com, 2:30 pm,          Lacy Marshall
```

Although the comma is used in the above example as the delimiter character, the semicolon, tab, carrot ^, or tilde ~ could be used instead.

Pass a NULL or empty string for ListPathName to rewind the recipient list file. This allows a second pass through the list to send email when a first pass was a "merge-only" pass.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : List file successfully opened.

EXAMPLE CODE

```
ListFile = "c:\mcm4db\apps\list.txt"
Code = mcmOpenList(ListFile)
```

EXAMPLE PROGRAMS

SendMail

ALSO SEE

mcmOpenLetter

2.16 mcmOpenReply: Open reply file for processing.

SYNTAX

```
CLONG mcmOpenReply(CSTRING,CSTRING)

Code = mcmOpenReply(RemoveFile, RemoveString)

   RemoveFile   : (CSTRING) Pathname of reply file.
   RemoveString : (CSTRING) Reply string.
```

REMARKS

The **mcmOpenReply** function opens a reply file that is associated with the specified subject string. When reading client replies (to previously sent email), if the subject begins with
the specified string, as for example,

```
   REMOVE m.marshall0610@yahoo.com
```

then the email address following the string ("REMOVE" in the example above) is written to the reply file.

More than one reply file can be opened.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : No error.

EXAMPLE CODE

```
RemoveFile = "c:\mcm4db\apps\remove.txt"
RemoveString = "REMOVE"
* open file for replies on subject line "REMOVE email-address"
Code = mcmOpenReply(RemoveFile, RemoveString)
```

EXAMPLE PROGRAMS

GetReply

ALSO SEE

mcmOpenSkip and mcmOpenBounce

2.17 mcmOpenSkip: Open skip file for processing.

SYNTAX

```
CLONG mcmOpenSkip(CSTRING)

Code = mcmOpenSkip(SkipPathName)

   SkipPathName : (CSTRING) Pathname of skip file.
```

REMARKS

The **mcmOpenSkip** function opens a file containing email addresses of recipients to which email should not be sent, even if the email address appears in the list of recipients.

Typically, skip files are either a list of email addresses that were previously not deliverable or addresses of recipients who replied to previously sent email with one of the string specified in **mcmOpenReply**.

More than one skip file can be opened.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : No error

EXAMPLE CODE

```
BounceFile = "c:\mcm4db\apps\bounce.txt"

* Open file containing addresses of undeliverable email.
Code = mcmOpenSkip(BounceFile)
```

EXAMPLE PROGRAMS

SendMail

ALSO SEE

mcmOpenReply and mcmOpenBounce

2.18 mcmPop3Close: Close POP3 connection.

SYNTAX

```
CLONG mcmPop3Close(CVOID)

Code = mcmPop3Close()
```

REMARKS

The **mcmPop3Close** program closes the connection to the POP3 server.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : No error

EXAMPLE CODE

```
* close POP3 connection
mcmPop3Close()
```

EXAMPLE PROGRAMS

GetReply

ALSO SEE

mcmPop3Connect

2.19 mcmPop3Connect: Connect to POP3 server.

SYNTAX

```
CLONG mcmPop3Connect(CSTRING,CLONG,CSTRING,CSTRING)

Code = mcmPop3Connect(Server, Port, User, Pass )

   Server : (CSTRING)  POP3 server name or IP address.
   Port   : (CLONG)    POP3 port (normally 110).
   User   : (CSTRING)  POP3 user name.
   Pass   : (CSTRING)  POP3 password.
```

REMARKS

The **mcmPop3Connect** function connects to the specified POP3 server for the purpose of (1) reading replies from servers reporting than email was undeliverable and (2) reading replies from recipients.

Once connected, the number of messages in the POP3 account is returned.
Note: **mcmPop3Connect** and **mcmSmtpConnect** should not be called in the same program.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
= 0 : No messages on the server.
> 0 : The number of messages on the server.

EXAMPLE CODE

```
* connect to POP3 server
POP3_Server = "mail.hiwaay.net"
POP3_Port = 110
POP3_User = "username"
POP3_Pass = "secret"
Code = mcmPop3Connect(POP3_Server, POP3_Port, POP3_User, POP3_Pass)
```

EXAMPLE PROGRAMS

GetReply

ALSO SEE

mcmPop3Close

2.20 mcmReadReply: Read next email from POP3 server.

SYNTAX

```
CLONG mcmReadReply(CSTRING,CLONG,CLONG)

Code = mcmReadReply(SubjBuf, SubjLen, Flags )

   SubjBuf : (CSTRING) Reply buffer.
   SubjLen : (CLONG)   Size of buffer.
   Flags   : (CLONG)   Delete flags.
```

REMARKS

The **mcmReadReply** function reads the next email from the POP3 server, copying the subject to the 'UserBuf' buffer.

The email read is classified as one of three types:

(1) Email from servers indicating that the email was undeliverable.
(2) Email from recipients who have responded to one of the previous specified reply strings.
(3) All other email.

DeleteFlags specify if the email of the type specified in the above paragraph is to be deleted.

```
   MCM_DELETE_BOUNCED   1
   MCM_DELETE_MATCHED   2
   MCM_DELETE_OTHER     4
```

The above Delete Codes can be added together to expand the messages deleted. For example, to specify that only type 1 (bounced) and type 2 (recognized replied to) emails are to be deleted, but not others, set 'Flags' to 3. Hence, Flag = 3 will delete failure (bounced) email and recognized (matched) replies but keep all other messages. If no messages are to be deleted, use Flags = 0.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
= 0 : No reply string matches.
> 0 : Matched reply string (1 to 24).
= 999 : Email was undeliverable.

EXAMPLE CODE

```
DeleteCode = MCM_DELETE_BOUNCED + MCM_DELETE_MATCHED
SubjectBuffer = Space(256)
Code = mcmReadReply(SubjectBuffer, 255, DeleteCode)
```

EXAMPLE PROGRAMS

GetReply.

ALSO SEE

2.21 mcmRelease: Close down MCM.

SYNTAX

```
CLONG mcmRelease(CVOID)

Code = mcmRelease()
```

REMARKS

The **mcmRelease** function closes down all MarshallSoft Client Mailer (MCM) processing and should be the last MCM function called.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : No error

EXAMPLE CODE

```
mcmRelease()
```

EXAMPLE PROGRAMS

SendMail and GetReply

2.22 mcmSendMail: Sends merged mail.

SYNTAX

```
CLONG mcmSendMail(CVOID)

Code = mcmSendMail()
```

REMARKS

The **mcmSendMail** function sends the email created by calling **mcmMergeNext.**

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
> 0 : No error

EXAMPLE CODE

```
* send letter to recipient
Code = mcmSendMail()
```

EXAMPLE PROGRAMS

SendMail

2.23 mcmSetInteger : Sets numeric parameter for MCM processing.

SYNTAX

```
CLONG mcmSetInteger(CLONG,CLONG)

Code = mcmSetInteger(ParamName, ParamValue)

   ParamName  : (CLONG) Parameter number
   ParamValue : (CLONG) Parameter value.
```

REMARKS

The **mcmSetInteger** functions sets the specified integer parameter

MCM_ALLOW_EMPTY_FIELDS : Allows (1) or disallows (0) empty fields in the list of recipients, with the exception of the first field, which is reserved for the recipient's email address (since an email address must always be present). The default is 0 (empty fields not allowed).

MCM_SET_DEBUG_LEVEL : Changes the diagnostic debug level (initially set by **mcmAttach**) to MCM_DEBUG_OFF, MCM_DEBUG_LOW, or MCM_DEBUG_HIGH.

MCM_SET_DUPLICATE_DETECT : Enables (1) or disables (0) detection of duplicate email addresses in the recipient list. This does not affect operation of skip (exclusion) lists. The default is enabled (1).

MCM_SET_CHANNEL_DIVISOR : Sets the channel divisor D (default = 4) such that the number of channels N used is reduced so that (N <= L / D) where L = the number of lines in the recipient list. In order to take affect at runtime, **mcmOpenList** must be called before **mcmOpenLetter**.

MCM_SET_MACRO_DELIMITER : Specifies the macro substitution delimiter in the letter to be sent. Choose percent **%**, blackslash ****, or backquote **`** (the default).

MCM_AUTO_LOAD_HEADER_FILE : Sets a flag so that the header file will be automatically loaded when **mcmOpenLetter** is called, provided that the header file has the same name as the letter file except for extension ".hdr" rather than ".txt" or ".htm". Avoid having to call **mcmOpenHeader**..

MCM_SET_SMTP_PROTOCOL : Sets the SMTP protocol to 'ParamValue', which should be one of SMTP_AUTHENTICATE_CRAM, SMTP_AUTHENTICATE_LOGIN, or SMTP_AUTHENTICATE_PLAIN. Required by some SMTP servers. See \MCM4C\SSL\SSL_SERVERS.TXT.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
>= 0 : Parameter value set (no error).

EXAMPLE CODE

```
* set debug level to MCM_DEBUG_LOW
Code = mcmSetInteger(MCM_SET_DEBUG_LEVEL, MCM_DEBUG_LOW)
```

EXAMPLE PROGRAMS

None.

2.24 mcmSetProxySSL: Set SSL Proxy Parameters

SYNTAX

```
CLONG mcmSetProxySSL(CLONG,CLONG,CSTRING,CSTRING,CSTRING,CLONG)

Code =
mcmSetProxySSL(ProxyCode,ProxyFlags,ProxyDir,ProxyCert,ProxyExe,ProxyPort)

  ProxyCode   : proxy code (reserved, set to 0)
  ProxyFlags  : proxy server flags (1=icon on taskbar)
  ProxyDir    : proxy directory (on this machine)
  ProxyCert   : proxy certificate (STUNNEL.PEM) - file or path
  ProxyExe    : proxy executable (STUNNEL.EXE) - file or path
  ProxyPort   : proxy port
```

REMARKS

The **mcmSetProxySSL** program sets parameters for the proxy server (**Stunne**l) and must be called before connecting to any SMTP or POP3 server that requires SSL.

For details on using **Stunnel**, see the section "Using Stunnel" online in the MCM User's Manual or http://www.marshallsoft.com/stunnel.htm.

Set **`ProxyFlags = 1`** if an icon is to be placed on the task bar.
Set **`ProxyDir`** to the path used to write the **Stunnel** configuration and log files.
Set **`ProxyCert`** to the filename or pathname of the X509 certificate (in PEM format).
Set **`ProxyExe`** to the proxy executable filename or pathname.
Set **`ProxyPort`** to the proxy to be used to communicate with the proxy server, or 0 to disable the proxy server. Any unused port can be specified.

RETURNS

```
  < 0 : Error (see Section 3.0 MCM Error List)
>= 0 : No error.
```

EXAMPLE CODE

```
ProxyDir  = "c:\mcm4db\ssl"
ProxyCert = "c:\mcm4db\ssl\stunnel.pem"
ProxyExe  = "c:\mcm4db\ssl\stunnel.exe"

Code = mcmSetProxySSL(0,1,ProxyDir,ProxyCert,ProxyExe,8801)
```

EXAMPLE PROGRAMS

SendMail

2.25 mcmSetString: Sets string for MCM processing.

SYNTAX

```
CLONG mcmSetString(CLONG,CSTRING)

Code = mcmSetString(ParamName, ParamString)

   ParamName    : (CLONG)   Parameter number.
   ParamString  : (CSTRING) Parameter string.
```

REMARKS

The **mcmSetString** function sets a string parameter.

MCM_SET_FROM_ADDRESS : Sets the "From:" address on subsequent outgoing email (initially set by **mcmSmtpConnect**).

MCM_SET_CC_ADDRESS : Sets the "CC:" address string for all outgoing email. Addresses must be enclosed in '<' and '>' brackets, as in "<someone@comcast.net>".

MCM_SET_BCC_ADDRESS: Sets the "BCC:" address string for all outgoing email. Addresses must be enclosed in '<' and '>' brackets, as in "<someone@comcast.net>".

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
>= 0 : Length of parameter string (no error).

EXAMPLE CODE

```
FromAddr = "m.marshall0610@yahoo.com"
* set "From:" address
Code = mcmSetString(MCM_SET_FROM_ADDRESS, FromAddr)
```

EXAMPLE PROGRAMS

None.

2.26 mcmSleep: Sleeps specified milliseconds.

SYNTAX

```
CLONG mcmSleep(CLONG)

Code = mcmSleep(MilliSecs)

   MilliSecs : (CLONG) Milliseconds to sleep.
```

REMARKS

The **mcmSleep** function sleeps the number of specified milliseconds. This function is the same as the Windows API Sleep function.

RETURNS

MilliSecs

EXAMPLE CODE

```
* sleep 3 seconds
Code = mcmSleep(3000)
```

EXAMPLE PROGRAMS

SendMail and GetReply.

2.27 mcmSmtpClose: Close SMTP server connection.

SYNTAX

```
CLONG mcmSmtpClose(CVOID)

Code = mcmSmtpClose()
```

REMARKS

The **mcmSmtpClose** function closes all SMTP channels, and will not return until all channels are closed.

Before calling **mcmSmtpClose,** the Code = mcm**GetInteger**(**MCM_GET_CHANNEL_STATUS)** should be called repeatedly until it returns 0, indicating that all channels have finished sending. See the SendMail example program.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
>= 0 : No error.

EXAMPLE CODE

```
* close all connections to SMTP server
mcmSmtpClose()
```

EXAMPLE PROGRAMS

SendMail

ALSO SEE

mcmPop3Close

2.28 mcmSmtpConnect: Connect to SMTP server.

SYNTAX

```
CLONG mcmSmtpConnect(CSTRING,CLONG,CSTRING,CSTRING,CSTRING,CLONG,CLONG)

Code = mcmSmtpConnect(Server, Port, User, Pass, From, ReOpen, Delay)

   Server : (CSTRING) SMTP server name of IP address.
   Port   : (CLONG)   SMTP port (normally 25 or 587)
   User   : (CSTRING) SMTP user name (SMTP Authentication only)
   Pass   : (CSTRING) SMTP password (SMTP Authentication only)
   From   : (CSTRING) Email address of sender.
   ReOpen : (CLONG)   Number of emails before closing & reopening.
   Delay  : (CLONG)   Number of seconds to delay before reopening
                      connection.
```

REMARKS

The **mcmSmtpConnect** function connects to the specified SMTP server using the number of channels authorized by the MCM license, but not more than the maximum that was passed to the **mcmAttach** function.

The "well known port" for SMTP is 25. However, some servers require that port 587 be used, reserving port 25 only for other known SMTP servers.

If the SMTP server requires "SMTP Authentication", the user and password must be specified. Otherwise pass the empty string Chr(0).

The ReOpen value specifies the number of emails sent (by each channel) before closing and reopening the connection, and is normally used with servers that set a maximum number of emails that can be sent. Pass 0 to disable the ReOpen feature.

The Delay value is the number of seconds to delay after closing the connection (ReOpen > 0 was specified) before reopening it . Pass 0 to specify no delay.

RETURNS

< 0 : Error (see Section 3.0 MCM Error List)
>= 0 : No error.

EXAMPLE CODE

```
SMTP_Server = "mail.hiwaay.net"
SMTP_User = Chr(0)
SMTP_Pass = Chr(0)
SMTP_Port = 587
SMTP_From = "m.marshall0610@yahoo.com"

* connect to SMTP server on port 587
Code = mcmSmtpConnect(SMTP_Server, SMTP_Port, SMTP_User,
                      SMTP_Pass, SMTP_From, 0, 0)
```

EXAMPLE PROGRAMS

SendMail

2.29 mcmStatistics: Get runtime statistics.

SYNTAX

```
CLONG mcmStatistics(CLONG)

Code = mcmStatistics(ParamName)

   ParamName : Parameter number.
```

REMARKS

The **mcmStatistics** function returns the runtime statistic corresponding to 'ParamName' .

MCM_STAT_TOTAL_RECIPIENTS	:	Total number of recipients
MCM_STAT_BRACKETED_ADDRESSES	:	Number of bracketed addresses
MCM_STAT_INVALID_ADDRESSES	:	Number of invalid addresses
MCM_STAT_DUPLICATE_ADDRESSES	:	Number of duplicate addresses
MCM_STAT_WITH_BAD_ATTACHMENT	:	Number of bad attachments
MCM_STAT_SKIPPED_ADDRESSES	:	Number of skipped addresses
MCM_STAT_WITH_UNKNOWN_CHARSETS	:	Number of unknown char sets
MCM_STAT_AVG_SEND_TIME	:	The average time (milliseconds) to send each email.
MCM_STAT_AVG_CONNECT_TIME	:	The average time (milliseconds) to connect to the server.

RETURNS

The selected runtime statistic.

EXAMPLE CODE

```
DupAddresses = mcmStatistics(MCM_STAT_DUPLICATE_ADDRESSES)
? Str(DupAddresses ) + " duplicate addresses seen"
```

EXAMPLE PROGRAMS

SendMail

ALSO SEE

mcmGetIntetger and mcmGetInterger2

2.30 mcmUtility: MCM Utility Function

SYNTAX

```
CLONG mcmUtility(CLONG, CSTRING)

   ParamName   : Parameter name.
   ParamString : Parameter string.
```

REMARKS

The **mcmUtility** function can only be called before calling any other MCM functions.

ParamName	ParamString	Returns
MCM_GET_FILE_LINE_COUNT	File name	# lines in file.

RETURNS

See above.

EXAMPLE CODE

```
Lines = mcmUtility(MCM_GET_FILE_LINE_COUNT, ListFile)
```

EXAMPLE PROGRAMS

SendMail.prg

2.31 mcmWriteToLog: Write to log file.

SYNTAX

```
CLONG mcmWriteToLog(CSTRING)

Code = mcmWriteToLog(Text)

   String : (CSTRING) Text to write to the log file.
```

REMARKS

The **mcmWriteToLog** function writes the specified string to the MCM log file. Note that **mcmWriteToLog** cannot be called until after **mcmAttach** is called.

RETURNS

The length of the passed string.

EXAMPLE CODE

```
Dim Text
Text = "(SendMail Example Program)"
mcmWriteToLog(Text)
```

EXAMPLE PROGRAMS

SendMail and GetReply

3.0 MCM Error List

The numerical list of MCM errors follows:

```
  -1: End-of-File (list)
-101: Cannot set SMTP port
-102: Cannot connect to SMTP server
-103: Invalid key code
-104: Send mail fails
-105: Cannot set SMTP user name
-106: Cannot set SMTP password
-107: Invalid email address
-121: Cannot connect to both SMTP and POP3
-141: Cannot set POP3 port
-142: Cannot connect to POP3 server
-171: Too many reply files
-201: Not authorized (internal error).
-202: First line of letter must start with 'To:'
-203: Second line of letter must start with 'Subject:'
-204: Body of email is missing
-205: Cannot open MCM bin-file
-206: Cannot read MCM bin-file
-207: Invalid bin-file format
-208: Corrupted bin-file
-210: Max recipient list size exceeded in evaluation version
-211: Cannot allocate memory for letter buffers
-212: TCP/TP running on Ethernet
-214: Path to MCM directory cannot be null
-215: Must specify path to MCM directory
-216: No such macro
-217: Maximum skip files exceeded
-218: Maximum reply files exceeded
-219: No such parameter
-220: No such header
-221: Unknown CharSet
-222: Buffer too small
-223: No channels allocated (by mcmAttach)
-224: All channels have been disabled
-225: Evaluation version expired
-226: String too long. Expect <= 256
-227: String too long. Expect <= 1000
-228: Bad email address. Expect '<name@domain>'
-229: Not authorized to use this version of MCM32.DLL
-230: File does not exist
-231: Email address should start with letter delimiter, not list delimiter
-232: Illegal letter delimiter. Expecting % \\ `
-233: Too many addresses. Limit is one address
-234: Brackets <..> not allowed in email addresses
-302: lstInit not called
-303: Cannot open list file
-304: No such string
-305: Bad delimiter. Expecting commas or tabs
-306: Cannot determine delimiter on macro line
-307: Cannot determine delimiter on entry line
-308: Delimiter character must match delimiter on macro line
-309: Number delimters must match number on macro line
-310: List buffer is too small
-311: Missing entry in recipient-list
-312: Recipient list string is too long
-351: Cannot start SMTP thread
-352: Maximum allowed channels exceeded
-353: No channels specified!
```

```
-354: mcmSmtpClose already called
-355: Not connected to servers
-401: End-of-File (letter)
-402: ltrInit not called
-403: Cannot open letter file
-404: Cannot allocate memory for (raw) letter
-405: Letter file not opened
-406: Macro not closed
-408: Macro too big
-409: Illegal character inside macro
-410: Macro not closed before end-of-line
-411: Isolated macro definition character (percent sign)
-412: Error reading letter file
-413: Macro cannot contain space characters
-414: Macro not found
-415: Unknown file extension: Expecting .htm, .txt, or .rtf
-451: Memory mutex operation failed
-452: Timed out waiting for memory mutex
-453: No such buffer exists
-454: bufInit not called
-455: Timed out waiting for free buffer
-501: Letter has not been loaded
-502: Macro not found in list macro line
-503: No such field in on list entry line
-504: Buffer overflow
-505: String table key too large (max = 40 chars)
-506: String table replacement text too large (max = 256 chars)
-507: String table overflow
-541: Supermacro not closed
-542: Illegal character in supermacro
-543: Supermacro too big
-544: Supermaccro table lookup fails
-545: Error reading INCLUDE file
-546: Include file too large (> 1024 chars)
-602: logInit not called
-701: Max files exceeded
-702: Cannot allocate memory
-703: No such file
-704: No such file index
-705: String not found
-801: No space remaining in file table
-802: No such file (bad file table entry)
-803: File not open
-902: Cannot allocate memory
-903: Slot table overflow
-951: MCM aborted
-952: Bad key code
-953: Evaluation version exported
-954: Bad edition code
-955: Must call mcmAttach first
```